Associating MAC addresses with addresses in a look-up table

Field of the invention

The present invention relates to methods of associating MAC addresses with addresses in a look-up table. The invention further relates to a switch such as an ethernet switch employing the method.

Background of the invention

Each computer on a LAN (local area network), such as an ethernet LAN, has a unique address called a MAC (media access control) address. An Internet switch which is connected to various computers of the LAN through its different respective switch ports, needs to learn the MAC address associated with each switch port, and may additionally store other data pertaining to each of these MAC addresses. It does this by defining a look-up table in RAM, so that each MAC address is associated with an address in the look-up table (here referred to as a "look-up table address"). The look-up table needs to store at each address the correspondence data for each MAC address which becomes associated with that address. This correspondence data includes the MAC address itself, and also data which is to be stored about that MAC address (e.g. the port with which the computer having that MAC address is associated). The MAC address alone is 48-bits wide, so a large RAM is needed to define the look-up table.

The association of MAC addresses with addresses in the look-up table proceeds by an automatic algorithm. Clearly, the RAM cannot be so large as to have a number of look-up table addresses equal to the total number of possible MAC addresses (i.e. 2 to the power of 48). Therefore, a correspondence is defined between MAC addresses and memory location, and for this a process known as "hashing" is used. "Hashing" refers to the

process of mapping the 48-bit MAC address to a shorter look-up table address using a compression algorithm, such as a Cyclic Redundancy Code (CRC) algorithm to reduce the 48-bit addresses to X bits, where X is the number of bits defining an address in the look-up table. Typically X is 9 or 10.

Often more than one of the computers attached to a single switch will be mapped by the hashing to the same look-up table address. Hence, the RAM will have to have a large enough memory capacity to store, at each look-up table address, all the correspondence data which may have to be stored for each of the MAC addresses which might be mapped to that look-up table address. In other words, the memory requires a large "width" for each look-up table address. This is despite the fact that, while there may be a few addresses in the look-up table which are mapped to several MACs of the LAN, there will typically be a very large number of look-up table addresses which are not mapped to any MAC addresses. In other words, there is an inefficiency. The implication is that there is an unnecessary increase of the memory size. A possible solution to this problem is provided by using content-addressable memory, but this is both costly and complex, so a simpler solution is desirable.

Summary of the invention

The present invention seeks to alleviate at least partially the problems described above.

In general terms, the present invention proposes that when, as the look-up table is constructed, a given MAC address hashes to a look-up table address which is already occupied (i.e. there is already a MAC address associated with that look-up table address), the MAC address is re-hashed to provide a different look-up table address. This procedure can be performed

any number of times. If it is performed sufficiently frequently, then it is likely that an address will be found which is free. In this way, the number of MAC addresses which will typically have to be associated with a single look-up table address is reduced, preferably to only one.

Specifically, a first expression of the invention is a method of associating look-up table addresses with MAC addresses, the method including for successive MAC addresses A_0:

using A_0 to generate y+1 look-up table addresses H_0, H_1, H_2, ..., H_y, where y is an integer greater than or equal to one; and

according to at least one criterion, associating the address A_0 with a selected one of the addresses H_0, H_1, H_2, ..., H_y.

The criterion may, for example, be that A_0 is associated with H_n where n is the smallest integer in the range 0 to y such that there is presently no MAC address associated with the address H_n, or more generally such that the number of MAC addresses presently associated with the address H_n is less than a predetermined integer.

The addresses H_1 to H_y are preferably generated successively, upon it being determined that the previously generated address fails to meet the criterion. For example, H_{n+1} may be generated only in the case that it is found that H_n does not meet the criterion.

The value of y may be predetermined, such that the maximum number of addresses H_0, H_1, H_2, ..., H_y which are generated is no more than a predetermined integer, even if none of these addresses meets the criterion. In this case a second criterion may be used to select which of the addresses H_0, H_1, H_2, ..., H_y is associated with the address A_0.

Alternatively, the value of y may be unlimited, and the method may generate addresses continually until at least one is found which meets the criterion.

Each of the addresses H_1, H_2, ..., H_y is preferably obtained from the address A_0 by the following steps. Firstly, we forming a respective string S_n

having the same number of bits as A_0 (according to present technology, 48). These S_n may just be respective sections of A_0 and in this case we optionally select one S_n (say S_1) and XOR it component-by-component with each of the other y-1 S_n, so that each of the other y-1 S_n is modified. Then each S_n (or modified S_n) is modulated with a respective set of Walsh codes (of the kind widely used in CDMA encoding for example). The y resultant strings are used in the same CRC which transformed A_0 to H_0, to produce H_n. Due to the use of Walsh codes, the likelihood is higher of the H_n for different MAC addresses A_0 being different from each other.

In a second aspect, the invention provides an Ethernet switch which performs a method according to the invention.

Specifically, this aspect of the invention may be expressed as an switch including a memory for defining a look-up table having a plurality of addresses and a processor for associating MAC addresses with addresses of the look-up table,

the processor being arranged to use each MAC address A_0 to generate y+1 look-up table addresses $H_0, H_1, H_2, \ldots, H_y$ for y an integer greater than or equal to one, and according to at least one criterion to associate the address A_0 with a selected one of the addresses $H_0, H_1, H_2, \ldots, H_y$.

Naturally, the various preferred features of the method are also preferred features of the switch.

Brief description of the figures

An embodiment of the invention will now be described in detail for the sake of example only, with reference to the following figures in which:

Fig. 1 shows schematically the ways the embodiment uses a 48-bit MAC address to form four different look-up table addresses;

Fig. 2 shows an algorithm for constructing a look-up table in the embodiment; and

Fig. 3 shows an algorithm for retrieving data from a look-up table formed by the algorithm of Fig. 2.

Detailed Description of the Embodiment

The method used by the embodiment to generate multiple look-up table addresses from a single MAC code is illustrated schematically in Fig. 1. The 48-bit MAC address is there called A_0. A_0 can be hashed by a known CRC to form an address of any desired number of bits (typically 9 or 10 bits). The MAC address for A_0 generated in this way is referred to here as look-up table address H_0.

The embodiment proposes that 3 alternative look-up table addresses may be created. The first is formed from the first 16 bits of the 48-bit MAC address, S_1. The second is formed from the second 16 bits of the 48-bit MAC address, S_2. The third is formed from the final 16 bits of the 48-bit MAC address, S_3. Generally, these 16-bit strings are referred to here as S_n, for integer n=1,..., 3. S_2 and S_3 are then preferably modified by XORing them, component-by-component with S_1.

Each of the 16-bit strings S_n is then used to generate a corresponding 48 bit string A_n, n=1,..., 3 by spreading/modulating the corresponding string S_n by using a respective code which is formed as a 16-bit concatenation of 3 different 16-bit Walsh codes. The nine 16-bit Walsh codes are written $W_{n,m}$ n=1,...3, m=1,...3. The first three 3 components of A_n are formed by an XOR of the first component of S_n with the first three components of $W_{n,1}$ respectively. Similarly, the second three components of A_n are formed by an XOR of the second component of S_n with the second three components of $W_{n,1}$ respectively. And so on. The sixth three components of A_n are formed by XORing the sixth component of S_n by the last component of $W_{n,1}$ and the first two components of $W_{n,2}$. And so on.

The same CRC is then used to generate a look-up table address H_n for each of these strings A_n.

The algorithm by which a new MAC address A_0 is added to the look-up table is illustrated in Fig. 2, and has the following steps.

In step 1, the address A_0 is received.

In step 2, the address A_0 is hashed by the CRC to form look-up table address H_0, and the integer variable n is set to 0.

In step 3, it is determined if the look-up table address H_n is already occupied. If the answer is "no", then the MAC address A_0 can be associated with the address H_n and the algorithm terminates.

If the answer at step 3 is "yes", the algorithm determines in step 4 if n is less than 3.

If the answer is "yes", then in step 5 the algorithm increases n by 1, forms H_n, and then returns to step 3.

If the answer is "no" then the algorithm terminates. No free addresses have been found at any of H_0, H_1, H_2, or H_3. In this case, the algorithm finds the one of the addresses out of H_0, H_1, H_2 and H_3 for which the association with its MAC address was formed furthest into the past, deletes this association, and associates A_0 with that address. In this way, as new MAC addresses are received each new address always becomes associated with a look-up table address which is not presently occupied, but sometimes old MAC addresses lose their association with any entry of the look-up table.

Fig. 3 shows an algorithm for extracting information about a certain MAC address from a look-up table generated by the embodiment.

In a first step 11 a MAC address A_0 is received.

In step 12 the integer variable n is set to 0, and the same CRC is used to generate a first look-up table address H_n.

In step 13, it is determined whether the look-up table address H_n is associated with the MAC address A_0 (this can be done by examining the correspondence data at address H_n in the look-up table). If the answer is "yes"

then the required information is extracted from the address H_n, and the algorithm terminates.

If the answer is "no", the algorithm proceeds to step 14 in which it is verified whether n is less than 3. If no, then the algorithm has failed to find any look-up table address associated with A_0. In this case, the system may proceed in any of the ways which are known in the prior art in comparable circumstances. For example, if information is to be transmitted to a computer with the MAC address A_0, that information may be multicast (i.e. transmitted through a group of the ports) or broadcast (i.e. transmitted through all of the ports), in order that it should reach that computer.

If the answer is "yes" the algorithm proceeds to step 15 in which n is set to n+1, $W_{n,1}$, $W_{n,2}$ and $W_{n,3}$ are used to generate A_n, and the CRC is used to generate look-up table address H_n from A_n. After this the algorithm returns to step 13.

Although the invention has been explained above with reference to a single embodiment. Many variations of this algorithm are possible within the scope of the invention as will be clear to a skilled reader.

As a first example, although the algorithm has been shown trying just four look-up table addresses, this can be generalised to y addresses (i.e. the algorithm above illustrates the special case of y=3). There are various ways in which the 48-bit MAC address A_0 can be used to generate y different addresses H_n, n=1, ..., y as will be clear to a skilled reader. The only way in which Figs. 2 and 3 need be varied in this case is that the test at steps 4 and 14 becomes whether n is less than y.

As a second example, although the use of Walsh codes is preferred there are many ways in which a 16-bit string S_n can be converted into a 48-bit string A_n, and indeed many ways in which strings A_n can be generated without using strings S_n, as will be clear to a skilled reader.

As a third example, although the invention has been shown in Fig. 2 terminating when the number of look-up table addresses found to be occupied

is 3, in principle the algorithm may keep on generating new look-up table addresses until a certain criterion is fulfilled, e.g. that an unoccupied look-up table address is found with which the present MAC address can be associated.